Exhibit B

Johnson I text compared to
U.S. Patent No. 6,449,108
(Appl. No. 09/239,435)

REFERENCE TO PENDING APPLICATIONS

[0001] This application is a continuation-in-part of (a) U.S. patent application Ser. No. 09/239,425 entitled "A Secure Electronic Mail System" filed on Jan. 28, 1999 and (b) Ser. No. 09/255,837 entitled "Method For Information Encoding And Transfer" filed on Feb. 23, 1999 which are continuation-in-part applications of co-pending U.S. patent application Ser. No. 08/892,982, filed Jul. 15, 1997, and entitled "Combined Remote Access and Security System"; which is a continuation-in-part of U.S. patent application Ser. No. 08/752,249, filed Nov. 19, 1996, and entitled "Combined Remote Access and Security System".

REFERENCE TO MICROFICHE APPENDIX

[0002] This application is not referenced in any microfiche appendix.

TECHNICAL FIELD OF THE INVENTION

[0003] The invention relates generally to a data processing system, method and article of manufacture allowing for the dynamic reconfiguration of an input/output device controller. In particular, the present invention relates to a computer-based system, method and article of manufacture which supports and facilitates a remote configuration and utilization of an emulated input/output device controller via encrypted data communication between a plurality of users and said controller.

BACKGROUND OF THE INVENTION

~~1. Field of the Invention~~

~~The present invention is directed to an apparatus and method for a secure electronic mail communication system. More particularly, the invention is directed for use in communicating over networks where secure information exchange is required. The invention has utility in applications such as person-to-person communication over network systems, communications over the Internet, interbusiness network communications where security is required, and the like.~~

~~2. Prior Art~~

~~The use of keys for secure communications is well known. Secure communication systems, as well as key systems, are shown in U.S. Pat. No. 4,182,933, issued to Rosenblum on Jan. 8, 1980, entitled "Secure Communication System With Remote Key Setting"; U.S. Pat. No. 4,310,720, issued to Check, Jr. on Jan. 12, 1982; entitled "Computer Accessing System"; U.S. Pat. No. 4,578,531, issued to Everhart et al., on Mar. 25, 1986, entitled "Encryption System Key Distribution Method and Apparatus"; U.S. Pat. No. 4,965,804, issued to Trbovich et al. on Oct. 23, 1990, entitled "Key Management for Encrypted Packet Based Networks"; U.S. Pat. No. 5,204,961, issued to Barlow on Apr. 20, 1993, entitled "Computer Network Operating With Multi-Level~~

~~Hierarchial Security With Selectable Common Trust Realms and Corresponding Security Protocols"; and U.S. Pat. No. 5,416,842, issued to Aziz on May 16, 1995 entitled "Method and Apparatus For Key Management Scheme For Use With Internet Protocols At Site Firewalls".~~

~~U.S. Pat. No. 4,182,933, issued to Rosenblum on Jan. 8, 1980, discusses a "Secure Communication System With Remote Key Setting". The Rosenblum '933 patent describes a system wherein a first subscriber communicates with a key distribution center to get an updated key to initiate secure communications with a second subscriber. An overview of the system shows that the user dials a telephone number into the first subscribing unit. The first subscribing unit then places the telephone number into temporary memory storage. The first subscriber then retrieves its initial caller variable from memory and places it into a key generator. The first subscriber then retrieves the number of the key distribution center (KDC) from its memory and dials the number. Once a connection has been established the first subscriber sends its caller ID as well as the caller ID of the telephone number being called to the KDC. This information is not yet transmitted in a secure manner.~~

~~Once the KDC has received the information from the first subscriber, the KDC looks up the caller variable for both the first subscriber and for the telephone number being called. The KDC then generates a new caller variable for the first telephone number. The KDC then transmits the caller variable for the number being called, a new caller variable for the first subscriber, using a secure transmission controlled by the initial caller variable. If this transmission is successful, then the KDC will replace the old caller variable in its table format with a new caller variable and break the connection.~~

~~Once the first subscriber has received and deciphered the caller variable for the number to be called and its new key caller variable, it will replace the old and used initial caller variable key with the new caller variable key. The first subscriber will then send the key for the number to be called to the key generator, retrieve the telephone number to be called, and dial the telephone number. The first subscriber will then transmit any information input by the user to the second subscriber using the second subscriber key. The second subscriber will receive information that has been encoded with the second subscriber key and will decode the information and transfer it on to the second user. In an alternative embodiment, after the phone call between the first subscriber and second subscriber, the second subscriber will call and get a new key from the KDC. In this alternative embodiment, both the key for the first subscriber and for the second subscriber will be changed out on every telephone call.~~

~~U.S. Pat. No. 4,310,720, issued to Check, Jr. on Jan. 12, 1982 discloses a "Computer Accessing System". The specification discloses a method for communicating between an access unit and a computer. The user enters his password into an input device which is connected to an access unit. The access unit generates a pseudo random access key from the password that is entered. The access unit then sends the access unit number and the generated access key to the computer controller for access to the computer system. The computer controller receives the access unit number and access key. The computer~~

~~controller then verifies the access unit number. If the access unit number is properly verified, the computer controller will then compare the access code to the expected access code listed in a table in the computer's memory. This expected access code is generated using a congruent pseudo-random decoding algorthym. If the access key code and the expected code match, then the computer controller will establish a link between the access unit and the computer.~~

~~The access unit and the computer will talk through an encoded communication system. Both the access unit and the computer will use a randomly generated encryption key for encoding and decoding the communication. This key is independently generated by both the access unit and the computer and is not transmitted over the access unit to computer link. After the termination of the call between the access unit and the computer, the computer will generate and store the next access key number for that particular access unit.~~

~~U.S. Pat. No. 4,578,531 issued to Everhart et al. on Mar. 25, 1986 discloses an "Encryption System Key Distribution Method and Apparatus". This system allows the secure method for communication between a terminal "A" and terminal "B" by using a remote key distribution center. An initial signal is sent from terminal "A" to terminal "B" to initiate the process of generating a secure communication line. Terminal "A" then generates a new call set up key in preparation for communication with the key distribution center, and a partial session key which will be transmitted through the key distribution center to terminal "B". Terminal "A" then updates its verification information in preparation for communication with the key distribution center. Terminal "A" then initiates the connection with the key distribution center to which it sends its terminal address and the terminal "B" address and an encrypted message including the two generated keys and the verification information. At this point, terminal "A" will wait for the processing by the key distribution center.~~

~~The key distribution center will read the address information from the signal sent from terminal "A" and use this to access a de-cryption key previously sent in communication with terminal "A". The message from terminal "A" will then be de-crypted and the verification information will be updated. The key distribution center will then generate a bidirectional asymmetric encryption/de-cryption key pair. The first part of this key pair will be sent to terminal "A", and the second part of the key pair will be sent to terminal "B". A similar communication will happen with terminal "B".~~

~~The message to terminal "A" will consist of a subsequent call key for the next communication with a KDC, a partial session key which it received from terminal "B", verification information, and two other variables "Y" and "Q". These five pieces of information will be encrypted using the call set up key for the present communication with terminal "A" and the information will be transmitted to terminal "A". A similar encrypted message will also be sent to terminal "B" from the KDC.~~

~~Terminal "A" will de-crypt the message from the KDC and verify that the information is correct. Terminal "A" will then store the new communication key for the next~~

communication with the KDC, take down the channel to the KDC, and establish a communication channel with terminal "B". A similar process will happen at terminal "B". At this point, terminal "A" and "B" will be able to communicate securely using the partial keys that were exchanged through the KDC. Terminals "A" and "B" can then use a random number and the variables "Y" and "Q" to create a new key which may be used to communicate securely between terminals "A" and "B". By using the variables and a random number to generate a new communication key, a secure communication encryption message may be employed which cannot be known by any outsiders to terminal "A" and "B", including the KDC.

U.S. Pat. No. 4,965,804, issued to Trbovich et al., on Oct. 23, 1990, discloses a "Key Management For Encrypted Packet Based Networks". This method of key management uses a key distribution center for sending keys to remote locations so that a secure communication can be made. Specifically, the system is designed to be compatible with X.25 type packet switching networks. This compatibility requires a balanced transmission which is implemented by a transparent device between the source DTE and second YDTE. The source DTE sends a transmit request to the transparent device which responds with a dummy signal back to the source DTE. The transparent device then contacts the key management system and obtains a key. A similar key is sent to the transparent device for the second DTE. The transparent devices for the first DTE and the second DTE then establish a communication network with an encrypted signal transfer, and finally the source DTE talks to the second DTE through the transparent devices and the encrypted connection.

U.S. Pat. No. 5,204,961, issued to Barlow on Apr. 20, 1993, discloses a "Computer Network Rating With Multi-Level Hierarchial Security With Selectable Common Trust Realms and Corresponding Security Protocols". The invention involves a method for setting up network communications between two trusted computer systems. Each trusted computer has a common set of protocols for the protection of data contained therein. Thus, if a user for a trusted computer system attempts to send data to a non-trusted computer system, then the trusted computer system will stop the message transfer and will not allow the communication to occur. This system operates as a method for two trusted computers to talk over a network which is not physically secure against interlopers. Each computer that is a member of a specific trust realm enforces a predefined security policy and defines security levels for the data contained within the computer. Before a trusted computer transmits a specified message, the trusted computer checks the trust realm table to verify that both the transmitting and receiving computers are part of at least one common trust realm. If both computers are part of a common trust realm, then the message will be transferred using the appropriate protocol for that trust realm. If the computers are not both members of the trust realm, then the message will not be transmitted. The communication between two trusted computers consists of a message which is transmitted as a protocol data unit which includes a sealed version of the message, authenticated identifies for the sending system and user, the message security level label, and an identifier for the selected trust realm. The transmitted message is then received, processed for validity and if valid, the message is processed within the receiving computer.

~~U.S. Pat. No. 5,416,842, issued to Aziz on May 16, 1995, discloses a "Method and Apparatus For Key Management Scheme For Use With Internet Protocols at Site Firewalls". This system consists of separate private networks which communicate over an Internet type connection through firewalls. A private network "I" communicates through a firewall "A" to the Internet where the message is transferred to firewall "B" and then decoded and sent on to another private network "J". This allows private network "I" and private network "J" to communicate in a secure encapsulated message while having firewall protection. The invention begins with a source node "I" sending a data gram to the firewall "A". Firewall "A" has a secret value "SA" and a public value "PA". Similarly, firewall "B" is provided with a secret value "SB" and a public value "PB". In this manner both firewall "A" and firewall "B" can acquire a shared secret value "SAB" without having to communicate. The communication is initiated by providing firewall "A" and firewall "B" with initial values for all other secure firewalls on the network. Firewalls "A" and "B" then use secret value "SAB" to create a key "KAB". The transmitting firewall then generates a random key "KP" which is used to encrypt the received data. The key "KP" and the encrypted data are then all encrypted by the public key "KAB" for transmission over the Internet. Firewall "B" will then use key "KAB" to de-crypt the message for the private key "KP" and de-crypt the data that has been transmitted. In this manner the transmitting firewall can constantly be changing the private key "KP" which increases the security of the system.~~

~~The above-described key distribution and encryption systems suffer from the drawbacks of using known communication pathways, having known addresses, and some systems even transfer secure key information over the communication lines.~~

~~Hence, there is a need for an improved communication method which allows for encrypted information transfer to dynamic locations without transmitting the keys over the communication line.~~

[0004] The present invention provides for secured, real-time, configuration and utilization of an emulated input/output device controller. The instant invention advances the art by allowing its practice to be supported via an encrypted communications protocol interfacing with, and relying upon, the teachings, practices and claims disclosed in co-pending U.S. patent applications Ser. No. 09/239,425 and 09/255,837 (hereinafter synonymously referred to as "Secure Agent" or "SA").

[0005] Secure Agent Service Overview

[0006] The following overview is provided to facilitate a comprehensive understanding of the teachings of the instant invention. Secure Agent utilizes a secure login sequence wherein a client connects to a Secure Agent server using a key known to both systems and a client connects and presents the server with user identification (as used herein the term "client" refers synonymously to a remote user establishing, and communicating with the instant invention through Secure Agent allocation and encryption processes as taught in the above noted applications). If recognized, the Secure Agent server initiates a

protocol whereby the client's identification is verified and subsequent communication is conducted within a secured (encrypted) construct. For purposes of this overview, the term "server" should be considered a hardware configuration represented as a central processing unit wherein Secure Agent, a Host DLL and driver reside, and are executed. The term "DLL" as used herein refers to a Secure Agent host dynamically linked library (a.k.a. Host DLL). The term "DLL" or "dynamically linked library" is used in a manner consistent with that known to those skilled in the art. Specifically, the term "DLL" refers to a library of executable functions or data that can be used by a Windows application. As such, the instant invention provides for one or more particular functions and program access to such functions by creating a static or dynamic link to the DLL of reference, with "static links" remaining constant during program execution and "dynamic links" created by the program as needed.

[0007] The Secure Agent server presents a variable unit of data, such as the time of day, to the client as a challenge. The client must then encrypt that data and supply it back to the server. If the server is able to decrypt the data using the stored client's key so that the result matches the original unencrypted challenge data, the user is considered authenticated and the connection continue. The key is never passed between the two systems and is therefore never at risk of exposure.

[0008] The initial variable unit of data seeds the transmission of subsequent data so that the traffic for each client server session is unique. Further, each byte of data transmitted is influenced by the values of previously sent data. Therefore, the connection is secure across any communication passageway including public networks such as, but not limited to, the Internet. The distance between the client and server is not of consequence but is typically a remote connection. For accountability purposes, the actions of a client may be recorded (logged) to non-volatile storage at almost any detail level desired.

[0009] The access rights of each client (what the client is able to accomplish during a session) is governed by data stored on the Secure Agent server to which the client is associated. As an example, such rights might encompass the ability to administer and utilize the services of the server system, which would, in turn, include capabilities such as adding new client users, changing a user's rights, transferring new code to the server, using a feature (or service) of the server and more.

[0010] Consequently, Secure Agent allows for the transmission of new code to the server and for that code to be implemented upon demand by a client. Such dynamic, real-time implementation in turn, allows for the behavior of the server to be modified. It is to this behavior modification the instant invention addresses its teachings, and thereby advances the contemporary art.

[0011] As will be readily appreciated by those skilled in the art, though the instant invention utilizes encryption/decryption and code recognition technology associated with Secure Agent, an alternative technology may be employed in support of the instant invention without departing from the disclosure, teachings and claims ~~Additionally, there remains a need for a mechanism in which to log on to a computer system securely~~

~~without passing password.~~presented herein.

BRIEF SUMMARY OF THE INVENTION

~~In accordance with the present invention, an improved encoded or encrypted method for transferring information is provided which addresses the drawbacks of the prior art devices.~~

~~In accordance with one embodiment of the present invention a message is input to a first device which obtains a dynamic address from a first server to allow for connection to a second server.~~

~~A further embodiment of the invention allows for transmitting the message from the first device to the second server, receiving the message at the second server, storing the message until transfer to a second device as requested, and then transmitting the message to the second device from the second server.~~

~~Another embodiment of the present invention allows for encoding the message before it is input to the first device, and decoding the message after it has been received at the second device.~~

~~Yet another embodiment of the present invention allows for multiple servers which can be contacted to obtain the dynamic address of another server.~~

~~A still further embodiment of the present invention uses a remote administrator to control access both to the first server for obtaining the dynamic addresses, and to the second server for message transfers.~~

~~In accordance with another embodiment of the present invention, the user access to the secure name server is controlled by a remote administrator which creates, authorizes and deletes valid user ID/password combinations.~~

~~In accordance with another example of the present invention, the system allows for an electronic mail transfer between two users where a direct communication between the first user and second user never occurs. In this manner, two users can communicate without actually having a direct connection which is detectable by other parties.~~

~~The principal object of the present invention is to provide an easy to use, protected, electronic mail system for communication.~~

~~Another object of the present invention is to allow for the establishment of multiple electronic mail servers for different user categories.~~

~~A still further object of the present invention is to provide for a system which can communication on both secure and non-secure electronic mail servers.~~

~~Yet another object of the present invention is to provide for a program which allows for automatic and immediate deletion of electronic mail messages once they have been sent.~~

~~Other objects and further scope of the applicability of the present invention will become apparent from the detailed description to follow, taken in conjunction with the accompanying drawings wherein like parts are designated by like reference numerals.~~

~~DESCRIPTION OF THE DRAWINGS~~

[0012] The present invention is best viewed as comprised of two server components with one or more client subcomponents or sub-processes disclosed in association thereto. It can be further conceptualized that a distinguishable client component exists for each emulated device type recognized by the invention's server, with an individual client supporting the simultaneous use of a plurality of client-side components. As used throughout the instant invention specification and claims, the term "server" is used synonymously with "emulated device controller", "server central processing unit", "server CPU", and "remotely configurable input/output device controller" and the term "client" is used synonymously with "host user", "client central processing unit", "client CPU" and "remote user".

[0013] The invention's lower-most server component layer is a device driver to communicate directly with one or more hardware components attached to one or more computer systems, such as, but not limited to, mainframe computers (a.k.a. host processors). The driver controls the hardware in a manner prescribed by its design, causing it to interact with the other computer systems to which it is connected as if it were one or more device types (emulation). The driver additionally acts as a conduit to a higher level server component that governs the overall behavior of the emulated devices. This higher level component primarily supplies the driver with new data to provide through the emulated devices to the other computers to which it is connected and accepts data arriving to the emulated devices carried up by the device driver. Both layers predomoninantly operate on a device by device basis. The higher level server component, in turn, serves as the interface between Secure Agent technology and remotely connected clients allowing for the encrypted transmission of all data external to the server.

[0014] Using the example of an IBM 3215 console, a client would connect to a server and request a list of the 3215 devices which shared membership to the user's security groups. The user would select a device and a logical pathway from the mainframe computer to the client's system would become established. The client would communicate through the server layers with the end result of messages transported from a mainframe through an emulated device to the client for presentation within a window on a computer screen. Conversely, commands to the mainframe may be issued at the client's workstation and are transported through to the emulated device then through it to the mainframe.

[0015] Just as a client might have the ability to administer users (i.e. add/remove), a client might be able to modify the presence and behavior of emulated devices, via Secure

Agent administrative functions as taught by the afore noted pending patent applications. Allowable configuration ranges and values are verified and enforced according to rules by the server. The various data elements that may be controlled are listed at the bottom of this section. The server disallows modification of the active configuration (apart from device names and their security groups) and forces such modifications to be made to an inactive configuration. This inactive configuration may be swapped with the active configuration (thus activating it) upon demand. Thus, a new configuration may be prepared prior to a decision made to put it into effect. Additional control functionality includes but is not limited to the following:

[0016] Recycling an adaptor that is connected to an external computer system. This is commonly referred to as a Power On Reset or, more simply, a POR.

[0017] Viewing which users are connected to which devices.

[0018] Disconnecting a client user from a device to which he is connected.

[0019] Activating an inactive configuration.

[0020] Copying the active configuration to the inactive configuration in order to make changes based upon the active configuration.

[0021] Purging the inactive configuration in order to start fresh.

[0022] Consequently it is an object of the instant invention to provide for remote control, operation and use of a server Central Processing Unit (CPU).

[0023] A further object of the instant invention is to provide for a secured logon sequence utilizing encrypted data transmission in accordance with the teachings, disclosure and claims of the above noted pending patent applications.

[0024] Yet another object of the instant invention is to insure that all data transferred external of the emulated input/output device controller is encrypted in accordance with the teachings of the above noted pending patent applications.

[0025] A further object of the instant invention is to provide the ability for an administrator to alter and manage the configuration of emulated mainframe peripheral devices.

[0026] A further object of the instant invention is to allow the selective addition or restriction in the presence of devices to one or more host processors such as, but not limited to, mainframe computers.

[0027] Another object of the instant invention is to provide for a configuration specification which provides the ability to arbitrarily name each emulated device and assign it to one or more security groups of which a user must be a member in order to

access that particular device.

[0028] An additional object of the present invention is to provide the capability by which an administrator may add and remove one or more users with respect to emulated input/output device allocation.

[0029] Yet another object of the instant invention is to provide a facility by which an administrator may manage the security groups to which a user belongs, thus controlling the access of devices by users at any level desired down to an individual user level.

[0030] A further object of the instant invention is to provide the ability for a user to access and operate an emulated input/output device.

[0031] Yet another object of the instant invention is to provide the facility by which an administrator may effect/implement new device emulation types.

[0032] Another object of the instant invention is to provide support for multiple device types which may be simultaneously supported and operated.

[0033] Responsive to the foregoing challenges, the Applicant has developed an innovative system, method and article of manufacture to remotely configure and utilize an emulated device controller via an encrypted validation communication protocol.

[0034] It is to be understood that both the foregoing general description and the following detailed description are exemplary and explanatory only, and are not restrictive of the invention as claimed. The accompanying drawings, which are incorporated herein by reference, and which constitute a part of this specification, ~~FIG. 1 is a schematic view of a network communication arrangement utilizing a secure electronic mail system~~illustrate certain embodiments of the invention and, together with the detailed description, serve to explain the principles of the present invention.

~~FIG. 2 is a flow chart representation of the process to remotely administrate electronic mail accounts.~~

~~FIG. 3 is a flow chart representation of the process used to send mail.~~

~~FIG. 4 is a flow chart representation of a process used to retrieve mail.~~

~~FIG. 5 is a flow chart representation of a process to register a machine with a secure name server.~~

~~FIG. 6 is a flow chart representation of a process for obtaining an IP address from alternate secure name servers.~~

~~FIG. 7 is a flow chart representation of a process to get an IP address from a particular secure name server.~~

FIG. 8 is a flow chart representation of a connection process to a secure electronic mail server.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

In accordance with one exemplary embodiment of the present invention as shown in FIG. 1, a protected communication network is generally designated by the reference numeral 10.

In the preferred embodiment, the protected communication network 10 consists of a first central processing unit or user 12, a secure name server 14, a secure electronic mail server 16, a second central processing unit or user 18, a remote administrator 20 and a connecting network 22. The general operation of the overall system will be outlined in the following discussion.

Initially, the secure electronic mail server 16 will establish a link to a connecting network 22 and obtain a dynamic address. The dynamic address is standardly assigned by the network to a user of the network. An example of a dynamic address is a dynamic Internet protocol address for communicating over the Internet or world wide web. The secure electronic mail server 16 will then contact the secure name server 14 which has a fixed address on the connecting network 22. The secure electronic mail server 16 will then notify the secure name server 14 of the secure electronic mail server's 16 dynamic address on the connecting network 22. The communication between the secure electronic mail server 16 and the secure name server 14 will then be discontinued.

It will be understood that the present invention will be applicable to various types of networks.

Next, the remote administrator 20 will log on to the connecting network 22 and communicate with the secure name server 14. Note that this communication is a protected communication to allow for a protected information transfer. The secure name server 14 transfers the dynamic address of the secure electronic mail server 16 to the remote administrator 20. The communication between the secure name server 14 and the remote administrator 20 is then discontinued.

In an alternate embodiment, the remote administrator 20 will establish logon protocol for users to access the secure name server 14. The remote administrator 20 will then have the information to pass on to users of the protected communication network 10 to allow them to access the secure name server 14 through their logon protocol. In this manner, access to the secure name server 14 is controlled by the logon protocol, and only users authorized by the remote administrator 20 will be allowed to access the secure name server 14.

After receiving the dynamic address of the secure electronic mail server 16, the remote administrator 20 will initiate a communication with the secure electronic mail server 16

over the network 22. Once again, this is a protected information transfer communication. During this communication, the remote administrator 20 will create, change, and delete authorized user ID/password combinations for accessing the secure electronic mail server 16. The communication between the remote administrator 20 and the secure electronic mail server 16 will then be discontinued.

As different users require access to the system, the remote administrator 20 will provide the appropriate logon protocol and/or authorized ID/password combinations to the users to allow for access to the protected communication network 10. In this example, both the first user 12 and the second user 18 contact the remote administrator 20 for authorized logon protocol and user ID/password combinations.

The first user 12 now wishes to write and send an electronic mail communication to the second user 18 over the protected communication network 10. The first user 12 uses his unique logon protocol combination to access the secure name server 14 over the connecting network 22. Once again, this is a protected communication. The first user 12 then obtains the dynamic address of the secure electronic mail server 16 from the secure name server 14. The communication between the first user 12 and the secure name server 14 is then discontinued.

The first user 12 now uses his ID/password combination and the dynamic address to log onto the secure electronic mail server 16. Once the first user 12 has logged on to the secure electronic mail server 16, the first user's 12 electronic mail message is then protected by a protection method, such as encryption, and sent on the communication network 22 to the designated recipient's box on the secure electronic mail server 16. In this example, the information would be stored in the second user's box. The communication between the first user 12 and secure electronic mail server 16 is then broken.

At random intervals, the second user 18 will use his separate logon protocol to obtain the dynamic address of the electronic mail server 16 from the secure name server 14 and then access the secure electronic mail server 16 with his ID/Password combination to see if 20 there are messages for the second user 18. If there are messages in the second user's box on the secure mail server 16, the secure electronic mail server 16 will notify the second user 18 that there are messages available for retrieval. The secure electronic mail server 16 will then use a protected transfer to send the electronic mail message from the first user 12 to the second user 18 over the connecting network 22. The communication between the second user 18 and the secure electronic mail server 16 is then discontinued. Thus, a message has been transferred from the first user 12 to the second user 18 without a direct connection between the first user 12 and the second user 18.

It will also be understood that, in an alternate arrangement, the secure name server and the secure mail server may reside on the same computer system.

The aforementioned method of communication provides several levels of communication protection against outside interference for unwanted monitoring.

First, the first user 12 and the second user 18 never communicate directly. Thus, an outside person must monitor multiple communication pathways to detect communication between the first user 12 and the second user 18.

Second, because the secure electronic mail server uses a dynamic address, the communication pathways to and from the secure electronic mail server 16 are constantly changing. This increases the difficulty of monitoring communication with the secure electronic mail server 16.

Third, because the dynamic address of the secure electronic mail server 16 must be obtained from the secure name server 14, the address of the secure name server 14 must be known.

Fourth, because the secure name server 14 requires a proper log protocol combination, the dynamic address of the secure electronic mail server 16 is not easily obtained.

Fifth, because the secure name server 14 transfers the dynamic address of the secure electronic mail server 16 in an encrypted message, a first level of encryption must be broken just to obtain the dynamic address for the secure electronic mail server 16.

Sixth, because a communication between a user and the secure mail server 16 is protected, a second level of encryption must be broken to obtain the message.

Seventh, because the users can be using an additional protection or encryption system that is unknown to the secure networks, an additional level of protection can be used between the first user 12 and the second user 18. This additional level must also be broken to obtain the message text.

Eighth, because the entire system is controlled by a remote administrator 20, logon protocols, passwords, and keys can be constantly updated and changed. Any compromised logon protocol or ID/password combinations can be immediately deleted from the system by the remote administrator 20.

In addition, multiple applications of the present system could provide for a system where the communication between the remote administrator 20 and a secure electronic mail server 16 would also be an indirect communication through another electronic mail server 16.

While these descriptions of protection levels illustrate one example of the present invention, it is to be understood that the different levels of protection or additional levels of protection may be implemented in conjunction with the present invention to further enhance security.

The sub-processes for communicating throughout the network include the process to administrate electronic mail accounts, the process to send electronic mail, the process to

retrieve mail, the process to register a machine with a secure name server, the process to obtain a dynamic address from alternate secure name servers, the process to get an address from a secure name server, and the process to connect to a secure electronic mail server.

Each of the sub-processes for communicating will be given further detail in the following discussion.

Process to Administrate Electronic Mail Accounts

FIG. 2 of the drawings outlines the process by which the remote administrator sets up the user ID/password combinations. The process starts 30 by initializing the parameters necessary for operation of the process. The system will then check a first secure name server 32 for the dynamic address of the secure mail server. Block 34 represents the system checking to see it properly obtained the dynamic address of secure mail server from the first secure name server. If the system is successful in obtaining the secure mail server dynamic address from the first secure name server, the system will move on connect to the mail server as shown at block 36.

If the system is not successful in obtaining the dynamic address of the secure mail server from the first name server as shown in block 34, the system will move on to attempt to obtain the dynamic address of the secure mail server from the second secure name server, as shown in block 48. As shown in block 50, the system will check to see if it has now successfully retrieved the secure mail server dynamic address from the second secure name server. If the system is successful then the system will move on to connect to the secure mail server as shown in block 36. If the system has not successfully obtained the dynamic address of the secure mail server from either the first name server or the second secure name server the system will send back a report error as shown in block 52 and return an error code to the user as shown in block 54.

If the system has successfully obtained the dynamic address of the secure mail server, it will connect to the secure mail server using the dynamic address as shown in block 36. The remote administrator will then be able to add user ID/passwords as shown in block 38, modify user ID/passwords as shown in block 40, and delete user ID/passwords as shown in block 42. The remote administrator will then disconnect from the secure mail server as shown in block 44. The system will then end the process to remotely administrate as shown in block 46.

A similar process could be adapted to change the logon protocol for the secure name servers.

Process Used to Send Electronic Mail

FIG. 3 of the drawings outlines the process by which the secure electronic mail programs send mail communications. The process will start 60 by initializing the parameters necessary for operation of the process. The user will then use his logon protocol to check

a first secure name server 62 for the dynamic address of the secure mail server. Block 64 represents checking to see it properly obtained the dynamic address of secure mail server 20 from the first secure name server. If the user is successful in obtaining the secure mail server dynamic address from the first secure name server, the user will move on connect to the mail server at block 66.

If the system is not successful in obtaining the dynamic address of the secure mail server from the first name server as shown in block 64, the system will move on to get the dynamic address of the secure mail server from the second secure name server, as shown in block 74. As shown in block 76, the user will check to see if it has now successfully retrieved the secure mail server dynamic address from the second secure name server. If the user is successful, then the user will move on to connect to the secure mail server as shown in block 66. If the user has not successfully obtained the dynamic address of the secure mail server from either the first name server or the second secure name server, the user will send back the report error as shown in block 78 and return the error code to the operator as shown in block 80.

If the user has successfully used its logon protocol to obtain the dynamic address of the secure electronic mail server, it will connect to the secure mail server using the dynamic address as shown in block 66.

Once the user has successfully connected to the electronic mail server, the electronic mail is protected and sent to the electronic mail server as shown at block 68. The user then disconnects from the secure electronic mail server as shown at block 70, and ends the process as shown at block 72.

Process Used to Retrieve Mail

FIG. 4 of the drawings outlines the process by which a user retrieves mail from the secure mail server. The process will start 90 by initializing the parameters necessary for operation of the process. The user will use its logon protocol to check a first secure name server 92 for the dynamic address of the secure mail server. Block 94 represents the user checking to see it properly obtained the dynamic address of secure mail server from the first secure name server. If the user is successful in obtaining the secure mail server dynamic address from the first secure name server, the user will move on connect to the mail server at block 96.

If the user is not successful in obtaining the dynamic address of the secure mail server from the first name server as shown in block 94, the user will move on to get the dynamic address of the secure mail server from the second secure name server, as shown in block 110. As shown in block 112, the user will check to see if it has now successfully retrieved the secure mail server dynamic address from the second secure name server. If the user is successful, then the system will move on to connect to the secure mail server as shown in block 96. If the system has not successfully obtained the dynamic address of the secure mail server from either the first name server or the second secure name server, the user will send back the report error as shown in block 116 and return the error code to the user

as shown in block 118.

Once the user or retrieval program has properly connected to the electronic mail server, the electronic mail program will check to see if mail is available as shown in block 98.

If mail is available in block 98, then the retrieval program will retrieve the message headers as shown in block 100, retrieve the selected message as shown in block 102, delete the message from the secure mail server as shown in block 104, and disconnect from the secure electronic mail server as shown in block 106. The retrieval program will then restore the necessary parameters to properly end this process as shown in block 108.

If it is detected in block 98 that mail is not available, the retrieval program will disconnect from the secure mail server as shown in block 114.

Process to Register Machine with a Secure Name Server

As shown in FIG. 5, when a user, administrator, or secure electronic mail server logs onto the system with a dynamic address, the secure name server is contacted. The process for establishing this connection and supplying the proper dynamic address to the secure name server is outlined as follows.

As shown in block 120, the registering CPU machine selects an appropriate secure name server to be contacted. The registering machine then supplies the secure name server with these proper logon protocol combination as shown in block 122. As shown in block 124, a session with a secure name server is then established. If the session is successfully established as shown in block 126, then the machine will go on to register the dynamic address for the named machine 128, disconnect the session 130, and then properly shut down this process as shown in block 134.

If the session was not properly established in block 126, then the machine will report an error to the user or operator at block 136, and return an error code as shown in block 138.

Process to Obtain a Dynamic Address from Alternate Secure Name Servers

FIG. 2 of the drawings outlines the process by which a network user obtains a dynamic address from multiple secure name servers. The network user will use his logon protocol to check a first secure name server 140 for the dynamic address of the secure mail server. Block 141 represents the user checking to see it properly obtained the dynamic address of secure mail server from the first secure name server. If the user is successful in obtaining the secure mail server dynamic address from the first secure name server, the system will return the dynamic address to the user program as shown at block 142.

If the user is not successful in obtaining the dynamic address of the secure mail server from the first name server as shown in block 141, the user will move on to get the dynamic address of the secure mail server, from the second secure name server, as shown in block 143. As shown in block 144, the user will use its logon protocol to check to see

if it has now successfully retrieved the secure mail server dynamic address from the second secure name server. If the user is successful then the system will return the dynamic address to the user program as shown in block 142. If the user has not successfully obtained the dynamic address of the secure mail server from either the first name server or the second secure name server, the system will send back the report error as shown in block 145 and return the error code to the user as shown in block 146.

Process to Get an Address from a Secure Name Server

FIG. 7 of the drawings outlines the process by which an unknown address, such as the dynamic address of a secure mail server, is obtained from a secure name server. The process starts by selecting the target secure name server machine by its fixed address/name as shown in block 150. The user then provides the secure name server with its logon protocol combination as shown at block 152. If the user logon combination is verified then a session is established with a secure name server as shown at block 154. As shown at block 156, if the session has not been correctly established then the secure name server will report an error code as shown at block 178 and return the error code to the user as shown at block 180.

Returning to block 156, if the session has been correctly established as shown at block 156, then the user will be allowed to request the address for the named machine at the client site as shown at block 158.

The system will then perform a series of checks to see if the named machine has been properly identified. If the named machine has not been properly identified, shown at block 160, then the system will be disconnected as shown at block 172, move on to reporting the error code as shown at block 178, and continue processing.

If the named machine has been properly defined as shown at block 160, then the system will check to see if the named machine has properly registered its address shown at block 162. If the address has not been correctly registered, then the system will move on to disconnect session as shown at block 174, report the error code as shown at block 178, and continue processing. If the named machine has properly registered its address as shown at block 162, then the machine will check to see if the registration is up to date as shown at block 164.

If the registration is not properly up to date as shown at block 164, then the system will disconnect the session as shown at block 176, move on to report the error code as shown at block 178, and continue processing.

If the system registration has been properly updated as shown at block 164, then the system will return the obtained address as shown in block 168 and disconnect the session as shown in block 166. The system will then end processing as shown at block 170.

Process to Connect to Secure Electronic Mail Server

~~FIG. 8 of the drawings outlines the process by which a connection to a secure electronic mail server is made. The process begins by the user selecting the secure electronic mail server using the current dynamic address as shown at block 190. The user will then provide the user ID/password combination for the target secure mail server as shown at block 192. The user will then attempt to establish a session with secure electronic mail server as shown at block 194. The system will check to make sure that the session has been correctly established as shown at block 196.~~

~~If the session has been correctly established as shown at block 196, then the system will return to processing as shown at block 198 and allow the user to continue.~~

~~If the communication session has not been correctly established as shown at block 196, then the system will report an error as shown at block 200 and forward the error back to the user as shown at block 202.~~

~~The preferred embodiment of the present invention uses multiple secured name servers to allow for access to the secure mail server. However, it is also envisioned that a single secure name server or additional secure name servers could be used with this invention. It is also envisioned that the secure name server and the secure mail server could reside on the same machine. In this manner, two separate communication lines would be necessary to allow for the fixed address of the secure name server while providing for a dynamic address of the secure mail server.~~

~~It is also envisioned that the logon combination and user ID/password combination could be identical.~~

~~While the foregoing detailed description has described several embodiments of the secure electronic mail system in accordance with this invention, it is to be understood that the above description is illustrative and not limiting of the disclosed invention.~~

~~The claims and the specification describe the invention presented and the terms that are employed in the claims draw their meaning from the use of such terms in the specification. The same terms employed in the prior art may be broader in meaning than specifically employed herein. Whenever there is a question between the broader definition of such terms used in the prior art and the more specific use of the terms herein, the more specific meaning is meant.~~

~~While the invention has been described with a certain degree of particularity, it is manifest that many changes may be made in the details of construction and the arrangement of components without departing from the spirit and scope of this disclosure. It is understood that the invention is not limited to the embodiments set forth herein for purposes of exemplification, but is to be limited only by the scope of the attached claim or claims, including the full range of equivalency to which each element thereof is entitled.~~[0035] In this respect, before explaining at least one embodiment of the invention in detail, it is to be understood that the invention is not limited in this application to the details of construction and to the arrangement so the components set

forth in the following description or illustrated in the drawings. The invention is capable of other embodiments and of being practiced and carried out in various ways. Also, it is to be understood that the phraseology and terminology employed herein are for the purpose of description and should not be regarded as limiting. As such, those skilled in the art will appreciate that the conception, upon which this disclosure is based, may readily be utilized as a basis for the designing of other structures, methods and systems for carrying out the several purposes of the present invention. It is important, therefore that the claims be regarded as including such equivalent constructions insofar as they do not depart from the spirit and scope of the present invention.

[0036] Additional objects and advantages of the invention are set forth, in part, in the description which follows and, in part, will be apparent to one of ordinary skill in the art from the description and/or from the practice of the invention.

[0037] These together with other objects of the invention, along with the various features of novelty which characterize the invention, are pointed out with particularity in the claims annexed to and forming a part of this disclosure. For a better understanding of the invention, its operating advantages and the specific objects attained by its uses, reference would be had to the accompanying drawings, depictions and descriptive matter in which there is illustrated preferred embodiments and results of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

[0038] FIG. 1 is a system schematic providing a conceptual overview of primary hardware and software components of the instant invention as practiced in its preferred embodiment.

[0039] FIG. 2 is a logic flow diagram illustrating processing steps associated with the server initialization processing subcomponent of the instant invention when practiced in its preferred embodiment.

[0040] FIG. 3 is a logic flow diagram illustrating processing steps associated with the server termination processing subcomponent of the instant invention when practiced in its preferred embodiment.

[0041] FIG. 4 is a logic flow diagram illustrating processing steps associated with the adaptor configuration load processing subcomponent of the instant invention when practiced in its preferred embodiment.

[0042] FIG. 5 is a logic flow diagram illustrating processing steps associated with the client connection processing subcomponent of the instant invention when practiced in its preferred embodiment.

[0043] FIG. 6 is a logic flow diagram illustrating processing steps associated with the client disconnection processing subcomponent of the instant invention when practiced in its preferred embodiment.

[0044] FIG. 7 is a logic flow diagram illustrating processing steps associated with administrative functions given illustrative user response/input strings.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

[0045] While the making and using of various embodiments of the present invention are discussed in detail below, it should be appreciated that the present invention provides for inventive concepts capable of being embodied in a variety of specific contexts. The specific embodiments discussed herein are merely illustrative of specific manners in which to make and use the invention and are not to be interpreted as limiting the scope of the instant invention.

[0046] While the invention has been described with a certain degree of particularity, it is clear that many changes may be made in the details of construction and the arrangement of components without departing from the spirit and scope of this disclosure. It is understood that the invention is not limited to the embodiments set forth herein for purposes of exemplification, but is to be limited only by the scope of the attached claim or claims, including the full range of equivalency to which each element thereof is entitled. Turning now to FIG. 1.

[0047] In FIG. 1, a server CPU 103 has executing under control of its control program, Secure Agent software 106. The present invention advances the art and improves upon technology taught and claimed in the above noted pending applications, said applications and teachings incorporated by reference herein. The server 103 also has operating under control of its control program the remote configuration software 109 of the instant invention. Embodied within the server 103 is a hardware adaptor card 112. Said adaptor card 112 is in turn communicably attached to one or more host processors (121, 124, 127, 128). As used herein, the term "adaptor" refers synonymously to those hardware configurations such as, but not limited to, "adaptor cards" which allow for connectability between two or more central processing units and the transference of data associated therewith. Illustrative non-limiting examples of such adaptors as used herein would include Crossroads ESCON adaptors, Crossroads ESCON parallel adaptors, Bus-Tech adaptors and IBM ESCON adaptors.

[0048] In FIG. 1, the host processors (121, 124, 127, 128) are illustrated as Host 1 128 executing as its control program a VM system, Host 2 121 operating under as its control program a CICS system, Host 3 124 operating under the controller of its control program an IMS system and Host 4 127 operating under the dispatching control of its control program (ACP) a plurality of application specific programs. In turn, each of the host processors 128, 121, 124 and 127 illustrated in FIG. 1, have connected to it one or more physical input/output devices 131. In FIG. 1, said input/output devices are depicted as tape drives 141, direct access storage device 138 and smart terminals/personal computer/client computing capabilities 135. Also shown in FIG. 1 is a plurality of clients referred to as Host users 145 which are communicably attached to the server 103 of the instant invention via a communications network 148 such as, but not limited to, the

Internet or other computer compatible network wherein computer recognized and generated signals may be communicated between one or more central processing units.

[0049] Lastly shown in FIG. 1 is a Security Administrator client 151 interactively communicating with the Secure Agent software 106 operating within the server 103. As will be discussed in further detail and in association with FIGS. 2 through 7, the Security Administrator 151 utilizes Secure Agent software 106 to administer and maintain user/resource profiles 157 and further communicates with information conveyed to said Secure Agent software 106 via the software processes associated with the remote configuration software 109 of the instant invention.

[0050] For purposes of clarity and to assist in comprehension of the instant invention, it is convenient to view the invention as being comprised of a number of processing subcomponents. Such processing subcomponents include, but are not limited to, Server Initialization Server Termination, Adaptor Configuration Load, Client Communication, Client Termination and Administration related subcomponents.

[0051] The following discussion in association with FIG. 1 provides a brief non-limiting synopsis of the teachings of the instant invention and generally discusses the interrelationships of hardware and software processing components of the instant invention. In FIG. 1, a Security Administrator 151 defines via Secure Agent software 106, user and resource profiles 157. Such profiles are stored in a non-volatile storage medium, such as but not limited to, a disk drive 158. User resource records are those records which typically define security group or groups, and access control variables associated with the user. Stated succinctly, the user resource record/profile defines those resources that the user may utilize and the bounds of such utilization. The Security Administrator 151 may also define resource profiles, such resource profiles define the device type and grouping of emulated input/output devices as well as central processing unit designations associated with each emulated device type and/or grouping. When attempting to establish a session between a host user 145 and any one of the operating systems and/or application programs operating under the dispatching control of the operating systems of host processors 128, 121, 124 or 127, a user via a communications network 148 communicates first with Secure Agent software 106 operating within the server 103 of the instant invention 109. Assuming the user 145 is recognized as an authenticated and authorized user of the system as governed by Secure Agent software 106, the user 145 next requests a device or a device grouping of emulated input/output devices he or she anticipates utilizing in the requested session. The Secure Agent software 106 verifies the user 145 as authority to allocate such emulated input/output devices and correspondingly associates such devices with the user and user session between one or more of the host processors 128, 121, 124 and 127. Once established, the session continues as normal with input/output requests of the user serviced via emulated input/output device as opposed to the real input/output devices 131 associated with one or more of the host processors. Upon completion of the session or a specific deallocation request initiated by the user, the client termination subprocess of the instant invention deallocates the emulated input/output device or devices. As indicated, the processing subcomponents of the instant invention further include Adaptor Configuration Load,

Client Communication, Client Termination, Administration, Server Initialization and Server Termination subprocesses. It is to such subprocesses FIGS. 2 through 7 address themselves. A more detailed disclosure of each subprocess follows.

[0052] FIG. 2 discloses in further detail the process steps in which the server of the instant invention is initialized. While discussion of the individual subprocesses is provided in an illustrative logic sequence, it is to be noted that process steps defined therein need not occur in a serial manner. Rather it is expressly recognized that many of the subprocesses execution steps may be executed in a concurrent manner, or have their execution sequence factored upon the statusing of a previously executed process step.

[0053] Server Initialization, FIG. 2

[0054] With respect to server initialization, the driver of the instant invention first initializes all driver module-wide variables, such as clearing out how many adaptors are being supported, 201. Once these variables have been initialized, adaptors are located by enumerating all peripheral component interconnect computer Bus-type (PCI) devices present in the system using data and techniques published by the PCI Special Interest Group and by Microsoft's Window's NT Device Driver Kit (DDK). Specifically, the adaptor vendor and device IDs 202 are referenced to identify the presence of such supported adaptors. For each adaptor located, adaptor specific variables are initialized by the driver 203, with the resources used by the adaptor, such as buffer areas and IRQ (interrupt request lines) being next allocated and reserved 204 using functions provided by DDK. The adaptor is then reset 205 by the driver using a technique made known by the adaptor's manufacturer. Since these adaptors are generally intelligent it is necessary to transfer (download) to them microcode (a manufacturer-supplied program specific to such a device) that controls internal instruction sequencing. Therefore, microcode is downloaded into the adaptor 206 in a manner prescribed by the adaptor manufacturer with the adaptor then considered initialized 207. The driver next requests a connection to each unique IRQ so that any interrupts generated by any of the recognized adaptors may be serviced by the driver 208 and next initiates timer support 209 so that approximately once every second, general operations may be performed on behalf of each adaptor. This support typically, though not limitedly, includes ensuring an adaptor does not generate a non-detected interrupt. Having once initiated its timer 209, the driver next exposes standard module-wide support to all applications 210, which allows for communications with the driver as to be established by the Host DLL.

[0055] Subsequent to the driver initialization, the Host DLL initializes variables it utilizes 211 and clears a user connection block to allow information for each user to be represented 212. The Host DLL further exposes and makes available to Secure Agent a block of data, representing an emulated device specific administrative instruction set 213, for each user. In addition to such normal data elements as a user ID and password, this instruction set advises Secure Agent to maintain device type and security group strings on behalf of each user specifically for the support of this Host DLL. The device types limits those types of emulated devices to which a user might claim access whereas the device security groups name the emulated device security groups to which a user is subscribed.

In addition, at this stage linkage to configuration support routines within the Host DLL is also established. As practiced in one embodiment of the invention, the root name of the administrative tree structure is exposed to Secure Agent indicating that the Host DLL supports the configuration of information and will respond in a positive manner to requests for information and management of branches under this particular root. The Host DLL next creates a mutex serialization mechanism to be used by configuration support routines during access of adaptor configuration data to insure data integrity 214. This serialization mechanism is used to prevent for example potential simultaneous updates by multiple administrators as well as to prevent a client from enumerating emulated devices while it is being manipulated.

[0056] The Host DLL continues to open or otherwise establishes communication with the driver 215 and requests from it a number of recognized adaptors 216 to which the driver responds 217, whereupon the Host DLL requests from the driver its version number 218 to which the driver also responds 219. The Host DLL then records into a Secure Agent log the driver version and the number of adaptors it controls 220, and proceeds to indicate that each adaptor is not yet in a condition to support client connectivity 221. Data representing the adaptor configuration to be utilized (the active configuration) is next loaded 223. This data specifies device types and number of devices to be emulated, in conjunction with user-friendly (readable) names and security groups for each such emulated device. A second unique set of this data is loaded (the inactive configuration) 224 on behalf of this same adaptor to be used as a work area for administrators. This allows administrators to accumulate a series of configuration changes prior to effecting the activation of those changes as a whole. During said initialization, the Host DLL lastly ensures that the loaded adaptor configurations are within operationally permissible parameters 225.

[0057] FIG. 3 is a logic flow diagram illustrating processing steps associated with the server termination processing subcomponents of the instant invention as practiced in its preferred embodiment. Turning now to FIG. 3.

[0058] In FIG. 3 with respect to server termination, the Host DLL first disconnects each currently connected user 301. Such disconnection is facilitated via processing accommodated in the Client Disconnection Processing subcomponent as will be discussed in association with FIG. 6. Recognized adaptors are then set offline to their channels through the Adaptor Configuration Load processing subcomponent 302. The Host DLL next ceases communication, or closes the driver 303, and frees all allocated storage and resources 304. The one second timer is then closed by the driver 305 and module-wide exposure of support to application through NT is eliminated 306. The driver then ensures/verifies each adaptor is offline to the channel and the adaptor is reset 307, disconnects all previously connected IRQ's 308, and destroys each object instance 309. Such destruction further includes but is not limited to elimination of exposure of the emulated devices support to applications through NT 310 and the freeing of all allocated storage and resources 311.

[0059] FIG. 4 is a logic flow diagram illustrating the processing steps associated with the

Adaptor Configuration Load processing subcomponent of the instant invention as practiced in its preferred embodiment.

[0060] In FIG. 4 the Host DLL first indicates the adaptor's unavailability 401 and for each client currently connected to a logical unit on this adaptor, issues a message to the client indicating that the client is being disconnected due to administrative device management 402. The Host DLL then performs the client disconnection services in association with the invention's Client disconnection subprocess as will be discussed in further detail in association with FIG. 6. The Host DLL continues by next recording into Secure Agent log the configuration for this adaptor is being loaded 403 and if the adaptor is to be forced offline to the mainframe to which it is connected 404, prepare and uses an empty configuration indicating that Emulated devices are not to be emulated during this session. If the adaptor is not to be forced offline, an active configuration for the adaptor is provided and a request that the adaptor using the active configuration data is initiated 405. The driver as instructed causes the adaptor to be offline to the channel at this stage in the adaptor configuration load 406, destroys each of the adaptor emulated devices driver object instances 407 causing or eliminating the exposure of emulated devices support to applications through NT 408 and frees all allocated storage and resources 409. The driver next determines if Emulated devices are to be emulated 410 and then request that the adaptor be brought online to the channel 411, lastly indicating that the adaptor is available for client use 412.

[0061] FIG. 5 is a logic flow diagram illustrating the processing steps associated with the Client Connection processing subcomponent of the instant invention as practiced in its preferred embodiment.

[0062] Client Connection, FIG. 5

[0063] In FIG. 5, a client connection first initializes variables that it utilizes 501 then employs Secure Agent client code in order to establish a connection to the Host DLL 502, whereupon the Host DLL retains the client's name 503 and loads the client's device type and security groups 504. A new client object instance is then created to represent this new client connection with the variables it will use becoming initialized 505. The Host DLL then stores the location of the client object in a user connection block 506. At this point the client sends to the Host DLL the command version level that represents the client feature set as a means to facilitate backward compatibility by future Host DLLs 507 which the Host DLL stores for possible reference 508. By knowing the version of the client, the Host DLL can and will prevent communicating with older clients in a manner supported only by newer clients, whereas newer clients will be able to take advantage of a fuller set of features that the Host DLL offers. The client next provides to the Host DLL the emulated device type in which it is interested 509 whereupon the Host DLL stores it for later reference 510. The client then requests of the Host DLL its command version level 511 that the client stores for possible reference 513. Just as with the Host DLL being able to restrict its behavior for older clients, since the client knows the version level of the Host DLL it can restrict itself from attempting to take advantage of features available only on newer servers whereas newer servers might be more fully exploited.

The client then requests from the Host DLL a list of the currently available emulated devices to which the client may connect 514. The Host DLL returns the response back 515 whereupon the client selects one of the emulated devices and requests that the Host DLL establish a connection to it on its behalf 516.

[0064] FIG. 6 is a logic flow diagram illustrating the processing steps associated with Client Disconnection processing subcomponent of the instant invention as practiced in its preferred embodiment.

[0065] As can be seen in FIG. 6, the Host DLL destroys the client object instance which requires the following activity. If connected to a logical unit, the logical unit is closed 601 and the threads that were created to perform input/output device of the logical unit, if any, are terminated 602. If connected to a logical unit, the logical unit-in-use flag is set to not in use 603 and if connected to a logical unit, the logical unit client value is set to none 604. The Host DLL lastly frees all allocated storage and resources for the client object 605.

[0066] Administrative Configuration

[0067] When an administrator desires to modify the configuration of adaptors managed by the Host DLL it issues requests for enumeration of the "/Adaptors" root and its branches to which the Host DLL will respond. This provides the administrator with the means necessary to discover what information exists to be changed. The data exposed through these branches correlates to the data within the active and inactive configurations for each adaptor.

[0068] Once supplied with the name and value of a piece of adaptor configuration data an administrator can decide whether or not to make changes to it and, if so, supply that name with a new value back to the Host DLL which will then make that change on the administrator's behalf.

[0069] Additionally, an administrator may enumerate a series of controls that can be employed for special actions by the Host DLL against an adaptor. Specifically, an administrator might decide to activate the inactive configuration, whereupon the Host DLL will exchange the data of the active configuration with that of the inactive configuration then perform the actions detailed with Adaptor Configuration Load, FIG. 4. If, on the other hand, an administrator opted to copy the contents of the active configuration into that of the inactive configuration then the Host DLL would perform that action. An administrator also has the option to simply clear out the inactive configuration whereupon the Host DLL would reinitialize it to reflect the absence of configured emulated devices. If an administrator decided it was necessary to reinitialize the adaptor then he could specify that the Host DLL do so whereupon it would perform the actions detailed with Adaptor Configuration Load, FIG. 4. Finally, if an administrator decided that an adaptor should either be kept offline or could come back online then he could request that of the Host DLL and it would toggle that state for the adaptor then perform the actions detailed with Adaptor Configuration Load, FIG. 4.

[0070] Non-limiting examples of dialog and processing as provided for in the invention's administrative configuration subcomponent follow immediately for purposes of facilitating full and enabling disclosure.

[0071] Connected Client Traffic from Logical Unit: Mainframe Message (3215 Example)

[0072] When the adaptor interrupts with a message from the mainframe then that message is first caught by the driver emulated devices object and carried up into the Host DLL by a thread created on behalf of the client that performs I/O against the Logical unit. This message is then transmitted through SA to the client.

[0073] Connected Client Traffic from Logical Unit: Online or Offline Event (3215 Example)

[0074] When the adaptor is found to go online or offline to the channel then that event is first caught by the driver emulated devices object and carried up into the Host DLL by a thread created on behalf of the client that performs I/O against the Logical unit. This event is then transmitted through SA to the client.

[0075] Connected Client Traffic from Client: Mainframe Command (3215 Example)

[0076] The client may send a mainframe command to the Host DLL which is immediately transported to the driver emulated devices object by a thread created on behalf of the client that performs I/O against the Logical unit. The driver emulated devices object then requests that the adaptor send the command to the mainframe.

[0077] FIG. 7 is a logic flow diagram illustrating processing steps associated with administrative functions given non-limiting examples of user input command strings. Turning now to FIG. 7.

[0078] Administration of Adaptor Configuration Data: Input Request=Enumerate Branch /Adaptors

[0079] Administrator requests an enumeration of "/ESCON Adaptors" 701.

[0080] Host DLL builds and returns a string consisting of a concatenation of all the adaptors, in the form of Adaptor # where # is the 1-based number of the adaptor, along with a flag for each indicating that each element has, in turn, more branches 702.

[0081] Administration of Adaptor Configuration Data: Input Request=Enumerate Branch /Adaptors/Adaptor #

[0082] Administrator requests an enumeration of, for example, "/Adaptors/Adaptor 1" 703.

[0083] Host DLL builds and returns a string consisting of a concatenation of "Active Configuration" and "Inactive Configuration", each with a flag for each indicating that they have, in turn, more branches, along with a string of "Configuration Control" with a flag indicating that it has values 702.

[0084] Administration of Adaptor Configuration Data: Input Request=Enumerate Branch /Adaptors/Adaptor #/(In)Active Configuration

[0085] Administrator requests an enumeration of, for example, "/Adaptors/Adaptor 1/Active Configuration" 704.

[0086] Host DLL builds and returns a string consisting of a concatenation of 16 CUs, in the form of Control Unit x## where ## is hexadecimal from 00 through 0F, along with a flag for each indicating that each element has, in turn, more branches 702.

[0087] Administration of Adaptor Configuration Data: Input Request=Enumerate Branch /Adaptors/Adaptor #/(In)Active Configuration/Control Unit x##

[0088] Administrator requests an enumeration of, for example, "/Adaptors/Adaptor 1/Active Configuration/Control Unit x00" 705.

[0089] Host DLL builds and returns a string consisting of a concatenation of "Assignments" and "Logical Units", each with a flag indicating that they have values 702.

[0090] Administration of Adaptor Configuration Data: Input Request=Enumerate Branch /Adaptors/Adaptor #/(In)Active Configuration/Control Unit x##/Assignments

[0091] Administrator requests an enumeration of, for example, "/Adaptors/Adaptor 1/Inactive Configuration/Control Unit x00/Assignments" 706.

[0092] Host DLL builds and returns a string consisting of a concatenation of the following: 702

[0093] A. "Controller Type" with a flag indicating the data presentation to be a drop-down box.

[0094] This includes a list of all of the valid CUTypes (i.e. 7412, 3174) along with the currently assigned value. This value is taken from the specified Adaptor configuration data for this adaptor, indexed to the specified control unit.

[0095] B. "Base Address" with a flag indicating the data presentation to be a text box. This includes the currently assigned value. This value is taken from the specified Adaptor configuration data for this adaptor, indexed to the specified control unit.

[0096] C. "Device Count" with a flag indicating the data presentation to be a text box. This includes the currently assigned value. This value is taken from the specified Adaptor

configuration data for this adaptor, indexed to the specified control unit.

[0097] D. If the specified Adaptor configuration is the active configuration then a flag is added to all fields marking them as non-modifiable meaning that this data cannot be changed. For these particular datas only that within the inactive configuration may be worked upon.

[0098] Administration of Adaptor Configuration Data: Input Request=Enumerate Branch /Adaptors/Adaptor #/(In)Active Configuration/Control Unit x##/Logical Units

[0099] Administrator requests an enumeration of, for example, "/Adaptors/Adaptor 1/Inactive Configuration/Control Unit x00/Logical Units" 707.

[0100] Host DLL builds and returns a string consisting of a concatenation of the following: 702

[0101] A. For each emulated devices per Logical Unit Count for the specified Adaptor configuration data for this adaptor, indexed to the specified control unit (the following uses of ## is the current Logical Unit Count entry+the Logical Unit Base, providing the emulated devices address as it appears to the mainframe.):

[0102] 1. "Device x## Name(s)" with a flag indicating this is a text box. This includes the currently assigned value per the specified Adaptor configuration data for this adaptor, indexed to the specified CU and emulated devices per the current Logical Unit Count entry.

[0103] 2. "Device x## Group(s)" with a flag indicating this is a text box. This includes the currently assigned value per the specified Adaptor configuration data for this adaptor, indexed to the specified CU and emulated devices per the current Logical Unit Count entry.

[0104] 3. If the specified Adaptor configuration is the active configuration:

[0105] a. "Device x## Status" with a flag indicating this is a text box. This includes either the currently assigned emulated devices Client value (client userid) if the emulated devices In-Use flag indicates "in use", otherwise "this device is not in use". The emulated devices values involved are per the specified Adaptor configuration data for this adaptor, indexed to the specified CU and emulated devices per the current Logical Unit Count entry. This field is marked as non-modifiable meaning that this data cannot be changed (informational only)

[0106] Administration of Adaptor Configuration Data: Input Request=Enumerate Branch /Adaptors/Adaptor #/Configuration Control

[0107] Administrator requests an enumeration of, for example, "/Adaptors/Adaptor 1/Configuration Control" 708.

[0108] Host DLL builds and returns a string consisting of a concatenation of the following: 702

[0109] A. "Check this then click save to activate the inactive config" with a flag indicating this is a check box and a value of unchecked.

[0110] B. "Check this then click save to copy the inactive config to the inactive" with a flag indicating this is a check box and a value of unchecked.

[0111] C. "Check then then click save to purge the inactive config" with a flag indicating this is a check box and a value of unchecked.

[0112] D. "Check this then click save to POR the adaptor" with a flag indicating this is a check box and a value of unchecked.

[0113] E. "Force adaptor offline" with a flag indicating this is a check box. This includes the currently assigned value per the specified Adaptor configuration data for this adaptor.

[0114] Continuing with non-illustrated, non-limiting examples of Administrative processing functionality:

[0115] Administration of Adaptor Configuration Data: Data Assignment of a /Adaptors/Adaptor #/(In)Active Configuration/Control Unit x##/Logical Units value:

[0116] Administrator

[0117] 1. Requests an assignment of any modifiable value under "/Adaptors/Adaptor #/(In)Active Configuration/Control Unit x##/Logical Units" providing the new value along with the path to the data name.

[0118] Host DLL

[0119] 2. Assigns the specified data of the adaptor, indexed to the specified CU and Logical Unit, to the provided value.

[0120] 3. Saves the data to non-volatile storage through SA.

[0121] 4. If the change was to an emulated devices Name then, if that emulated devices is currently in use by a user per the emulated devices In-Use flag, use the emulated devices Client value to locate the client object then issue that client a message indicating the new emulated devices name.

[0122] 5. If the change was to an emulated devices Groups then, if that emulated devices is currently in use by a user per the emulated devices In-Use flag, use the emulated devices Client value to locate the client object and revalidate the client's authority exactly

as is in accordance with Client Connection discussion. If the client no longer has the authority to access the device then send him a message to that effect and perform Client Disconnection processing.

[0123] Administration of Adaptor Configuration Data: Data Assignment of a /Adaptors/Adaptor #/(In)Active Configuration/Control Unit x##/Assignments value:

[0124] 1. Administrator requests an assignment of any modifiable value under "/Adaptors/Adaptor #/(In)Active Configuration/Control Unit x##/Assignments", providing the new value along with the path to the data name.

[0125] Host DLL

[0126] 2. Ensures that every Logical Unit Base and Logical Unit Count is within the ranges established (and published) as acceptable to the adaptors and IBM mainframe computers. If not then reject the change

[0127] 3. Assigns the specified data of the adaptor, indexed to the specified CU, to the provided value.

[0128] 4. Saves the data to non-volatile storage through SA.

[0129] Administration of Adaptor Configuration Data: Admin checked /Adaptors/Adaptor #/Configuration Control/Check this then click save to activate the inactive config

[0130] 1. Administrator requests to activate the inactive configuration of the specified adaptor.

[0131] Host DLL

[0132] 2. Uses the configuration datas for the specified adaptor.

[0133] 3. Indicates that the adaptor is unavailable for use by clients.

[0134] 4. For each client currently connected to an emulated devices on this adaptor:

[0135] A. Issue a message to the client indicating that they are being disconnected due to administrator device management.

[0136] B. Perform Client Disconnection.

[0137] 5. Exchanges the contents of the active configuration with that of the inactive configuration.

[0138] 6. Saves the configurations to non-volatile storage through SA.

[0139] 7. Performs Adaptor Configuration Load.

[0140] Administration of Adaptor Configuration Data: Admin checked /Adaptors/Adaptor #/Configuration Control/Check this then click save to copy the inactive config to the inactive

[0141] 1. Administrator requests to copy the active configuration to the inactive configuration of the specified adaptor.

[0142] Host DLL

[0143] 2. Uses the configuration datas for the specified adaptor.

[0144] 3. Copies the contents of the active configuration into the inactive configuration.

[0145] 4. Saves the inactive configuration to non-volatile storage through SA.

[0146] Administration of Adaptor Configuration Data: Admin checked /Adaptors/Adaptor #/Configuration Control/Check then then click save to purge the inactive config

[0147] 1. Administrator requests to purge the inactive configuration of the specified adaptor.

[0148] Host DLL

[0149] 2. Uses the inactive configuration data for the specified adaptor.

[0150] 3. Clear it out to default values as does Start Server when a configuration doesn't exist. In summary, all of the CUTypes are assigned to 7412 and everything else is assigned to 0.

[0151] 4. Saves the inactive configuration to non-volatile storage through SA.

[0152] Administration of Adaptor Configuration Data: Admin checked /Adaptors/Adaptor #/Configuration Control/Check this then click save to POR the adaptor

[0153] Administrator requests to perform a Power On Reset (POR, or an offline/online recycle) of the specified adaptor.

[0154] Host DLL performs Adaptor Configuration Load for the specified adaptor.

[0155] Administration of Adaptor Configuration Data: Admin checked /Adaptors/Adaptor #/Configuration Control/Force adaptor offline

[0156] Administrator

[0157] 1. Requests a change to the flag that controls whether or not the adaptor is to be forced offline to the mainframe to which it is connected.

[0158] Host DLL

[0159] 2. Assigns the supplied setting to the data for the specified adaptor.

[0160] 3. Saves the value to non-volatile stored through SA.

[0161] 4. Performs Adaptor Configuration Load.

[0162] While this invention has been described to illustrative embodiments, this description is not to be construed in a limiting sense. Various modifications and combinations of the illustrative embodiments as well as other embodiments will be apparent to those skilled in the art upon referencing this disclosure. It is therefore intended that this disclosure encompass any such modifications or embodiments.

[0163] It will be apparent to those skilled in the art that various modifications and variations can be made in the construction, configuration, and/or operation of the present invention without departing from the scope or spirit of the invention. For example, in the embodiments mentioned above, variations in the materials used to make each element of the invention may vary without departing from the scope of the invention. Thus, it is intended that the present invention cover the modifications and variations of the invention provided they come within the scope of the appended claims and their equivalents.